A Flexible UMTS-WiMax Turbo Decoder Architecture

# A Flexible UMTS-WiMax Turbo Decoder Architecture

Maurizio Martina, *Member IEEE,* Mario Nicola, Guido Masera, *Senior Member IEEE*

***Abstract*—This work proposes a VLSI decoding architecture for concatenated convolutional codes. The novelty of this architecture is twofold: (i) the possibility to switch on-the-fly from the UMTS turbo decoder to the WiMax duo-binary turbo decoder with a limited resources overhead compared to a single mode WiMax architecture, and (ii) the design of a parallel, collision free WiMax decoder architecture. Compared to two single mode solutions, the proposed architecture achieves a complexity reduction of 17.1% and 27.3% in terms of logic and memory respectively. The proposed, flexible architecture has been characterized in terms of performance and complexity on a 0.13 $\mu$m standard cell technology, and sustains a maximum throughput of more than 70 Mb/s.**

***Index Terms*—VLSI, Turbo Decoder, UMTS, WiMax**

## I. Introduction

In the last years several standards have been proposed for reliable transmission of data over wireless channels (e.g. [1], [2]). Besides this, in order to cope with severe transmission environments, typical of wireless systems, channel codes ought to be adopted. Turbo codes [3] are among the most performing channel codes, and are still a major topic of interest in the scientific literature. Recent works dealing with turbo decoder implementation mainly focus on three aspects. **1)** The design of VLSI architectures for duo-binary turbo codes [4], [5], [6]. **2)** The design of flexible architectures able to support multiple codes [7], [8], [9]. **3)** The design of parallel decoders to sustain very high throughput (tens or hundreds of Mb/s), where the interleaver parallelization is particularly challenging, due to the problem of collisions in memory access [10], [11], [12]. Though current scaled CMOS technologies allow to reach clock frequencies of several hundreds of MHz, parallelization is still an effective methodology to achieve high throughputs and to approach the long term objective of 1 Gb/s in wireless communications. Furthermore, in high throughput ASIC design, the adoption of lower frequency parallel architectures instead of higher frequency serial ones is an effective method to combat unreliability and reduce nonrecurrent costs.

This work presents a high performance turbo decoder architecture, which faces the parallelization, the flexibility and the duo-binary implementation issues while keeping the complexity as reduced as possible, and achieves a throughput of several tens of Mb/s. Implementation results on a 0.13 $\mu$m standard cell technology show that the complexity overhead required to support both UMTS and WiMax is limited, compared to a single mode WiMax decoder architecture. Moreover, the proposed architecture yields noteworthy complexity reduction figures compared to a dual mode architecture, where no sharing technique is employed.

The rest of the paper is organized as follows. In section II the decoding algorithm is briefly recalled. Section III presents a reference architecture. In section IV the design of the low complexity interleaver employed in our architecture is addressed, whereas section V deals with the employed design strategies. Finally, section VI shows the experimental results and section VII draws some conclusions.

The authors are with Dipartimento di Elettronica - Politecnico di Torino - Italy. This work is partially supported by the MEADOW (MEsh ADaptive hOme Wireless nets) project, founded by the italian government. 

## II. Decoding Algorithms

Both the UMTS and the WiMax turbo codes are based on the parallel concatenation of two 8-state convolutional codes (CCs). However, the constituent code used in UMTS is a single binary systematic CC, whereas that used in WiMax is a duo-binary circular recursive systematic CC. At the decoder side, the SISO (Soft In Soft Out) module [13] executes the BCJR algorithm [14], usually in its logarithmic form [15]. Each SISO module receives the intrinsic log-likelihood ratios (LLRs) of coded symbols $c$ from the channel and outputs the LLRs of information symbols $u$. The two SISO modules exchange extrinsic LLRs ($\lambda_k[u]$) by means of interleaving memories $\Pi$ and $\Pi^{-1}$ (Fig. 1 (a)). The output extrinsic LLRs of symbol $u$ at the $k$-th step ($\lambda_k[u;O]$) are computed as:

$$\lambda_k[u;O] = \overset{*}{\max_{e:u(e)=u}}\{b(e)\} - \overset{*}{\max_{e:u(e)=\tilde{u}}}\{b(e)\} - \lambda_k[u;I] \quad (1)$$

where $\tilde{u}$ is an input symbol taken as a reference (usually $\tilde{u}=0$), $e$ represents a certain transition on the trellis and $u(e)$ is the uncoded symbol $u$ associated to $e$. The $\overset{*}{\max}\{x_i\}$ function [15] is implemented as a $\max$ followed by a correction term stored in a small Look-Up-Table (LUT) [16]. The correction term, usually adopted when decoding binary codes, can be omitted for duo-binary turbo codes [4].

It is worth pointing out that in binary turbo codes, at each trellis step the SISO outputs only one extrinsic LLR, whereas in duo-binary turbo codes the SISO produces three extrinsic LLRs; thus, in general, the terms $\lambda_k[u;O]$ and $\lambda_k[u;I]$ are vectors. The term $b(e)$ in (1) is defined as:

$$b(e) = \alpha_{k-1}[s^S(e)] + \gamma_k[e] + \beta_k[s^E(e)] \quad (2)$$

$$\alpha_k[s] = \overset{*}{\max_{e:s^E(e)=s}}\left\{\alpha_{k-1}[s^S(e)] + \gamma_k[e]\right\} \quad (3)$$

$$\beta_k[s] = \overset{*}{\max_{e:s^S(e)=s}}\left\{\beta_{k+1}[s^E(e)] + \gamma_k[e]\right\} \quad (4)$$

$$\gamma_k[e] = \pi_k[u(e);I] + \pi_k[c(e);I] \quad (5)$$

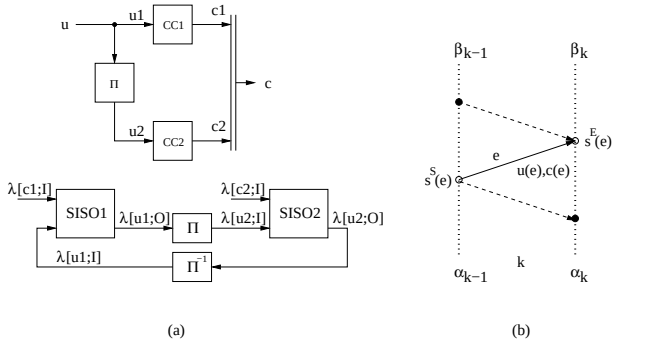


Figure 1. Parallel Concatenated Convolutional Codes: coder and iterative SISO based decoder (a), notation for the trellis section in the SISO (b)

where $s^S(e)$ and $s^E(e)$ are the starting and the ending states of $e$, $\alpha_k[s^S(e)]$ and $\beta_k[s^E(e)]$ are the forward and backward metrics associated to $s^S(e)$ and $s^E(e)$ respectively (see Fig. 1 (b)). The $\pi_k[c(e);I]$ term in (5) is computed as a weighted sum of the $\lambda_k[c;I]$ produced by the soft demodulator as

$$\pi_k[c(e);I] = \sum_i^{n_c} c_i(e)\lambda_k[c_i(e);I] \tag{6}$$

where $c_i(e)$ is one of the coded symbols associated to $e$ and $n_c$ is the number of bits forming a coded symbol. On the other hand, we can write $\pi_k[u(e);I] = u(e)\lambda_k[u(e)]$ for a binary turbo code, whereas for a duo-binary turbo code the $\pi_k[u(e);I]$ terms are piece wise functions:

$$\pi_k[u(e);I] = \begin{cases} 0 & \text{if } u(e) = (\text{`0'},\text{`0'}) \\ \lambda_k^{\overline{A}B}[u(e),I] & \text{if } u(e) = (\text{`0'},\text{`1'}) \\ \lambda_k^{A\overline{B}}[u(e),I] & \text{if } u(e) = (\text{`1'},\text{`0'}) \\ \lambda_k^{AB}[u(e),I] & \text{if } u(e) = (\text{`1'},\text{`1'}) \end{cases} \tag{7}$$

For further details on the theoretical aspects, the reader can refer to [13].

## III. Reference Architecture

The throughput of a turbo decoder ($T$), defined as the number of decoded bits ($N$) over the time required to perform the decoding operations ($D$), can be roughly estimated as

$$T = \frac{N}{D} = P\frac{Nf_{clk}}{2I(\tilde{N} + SISO_l P)} \tag{8}$$

where $P$ is the number of SISOs instanced into the decoder, $I$ is the number of iterations, $SISO_l$ is the SISO latency, $f_{clk}$ is the clock frequency and $\tilde{N}$ is equal to $N$ or $N/2$ for binary and duo-binary turbo decoders respectively. In a windowed architecture, the SISO latency is directly related to the window size $W$, as $W$ clock cycles are required for computing both the $\alpha$ and $\beta$ values. If boundary metrics calculated at previous iteration on the neighboring windows are used to initialize $\alpha$ and $\beta$ recursion, as suggested in [17], the SISO latency can be estimated as $SISO_l$=2$W$. Considering $W$=32 (typical value), $f_{clk}$=200MHz and $I$=8, a throughput $T \simeq 12$ Mb/s is obtained with $P = 1$. This value is more than sufficient to achieve the UMTS maximum throughput of 2Mb/s (block length $N = 5114$). On the other hand the WiMax standard requires a throughput close to 70 Mb/s for the maximum block length $N_c = N/2 = 2400$. Considering the same parameters listed above for UMTS, we obtain $P \geq 4$. As a consequence, the WiMax turbo decoder ought to be implemented as a parallel architecture, for the same clock frequency $f_{clk}$=200MHz.

A large part of the decoder area is devoted to the interleaver memory and the SISO modules, so these blocks can be effectively shared between the two decoders. The general SISO architecture is shown in Fig. 3. The $\alpha$ processor implements (3) and the $\beta$ processor implements (4) on two consecutive windows of data. Both the $\alpha$ and $\beta$ processors compute in parallel all the new State Metrics (SMs) (see the SM proc. block in Fig. 3). Since the $\alpha$ processor works in direct order on the input data, whereas the $\beta$ processor elaborates them in reverse order, two Branch Metrics Units (BMUs), are placed before the $\alpha$ and $\beta$ processors. Each BMU is devoted to combine $\lambda_k[u;I]$ and $\lambda_k[c;I]$ and obtains in parallel the Branch Metrics (BMs) $\gamma_k$ associated to the $k$-th trellis section. As a consequence a local buffer (BMU-MEM) is required to store $W$ extrinsic information and channel transition information values. The $\lambda - O$ processor generates the $\lambda_k[u;O]$ values according to (1) receiving the $\beta_k$ values directly from the $\beta$ processor and loading the $\alpha_k$ values from a local buffer ($\alpha$-MEM).

It is known that in binary turbo code decoders LLRs are commonly used. On the contrary, in duo-binary turbo decoders the use of logarithmic probabilities (LP) instead of LLRs allows to save a certain amount of logic in the SISO architecture [5]. However, the use of 4 LPs instead of 3 LLRs has a negative impact on both the interleaver memory and the BMU-MEM footprint. In order to select the most suitable approach, we implemented both the LLR based SISO (SISO-LLR) and the logarithmic probabilities based SISO (SISO-LP) in VHDL and synthesized them on a $0.13\mu$m standard cell technology. Moreover, we generated the dual port SRAMs to implement the interleaver memory both for the SISO-LLR (2p-LLR) and the SISO-LP (2p-LP) and the single port SRAMs to implement the BMU-MEM as a "ping-pong" buffer for both the cases (1p-LLR and 1p-LP). Fig. 2 shows the complexity

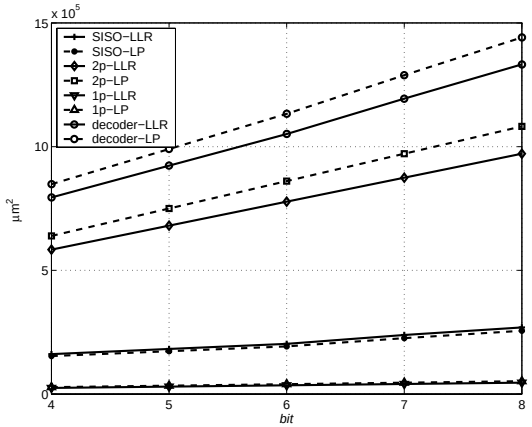


Figure 2. Complexity growth [$\mu$m$^2$] of a duo-binary turbo decoder building blocks as a function of the number of bits (*bit*) to represent the input data

growth of SISO-LLR, SISO-LP, 2p-LLR, 2p-LP, 1p-LLR and

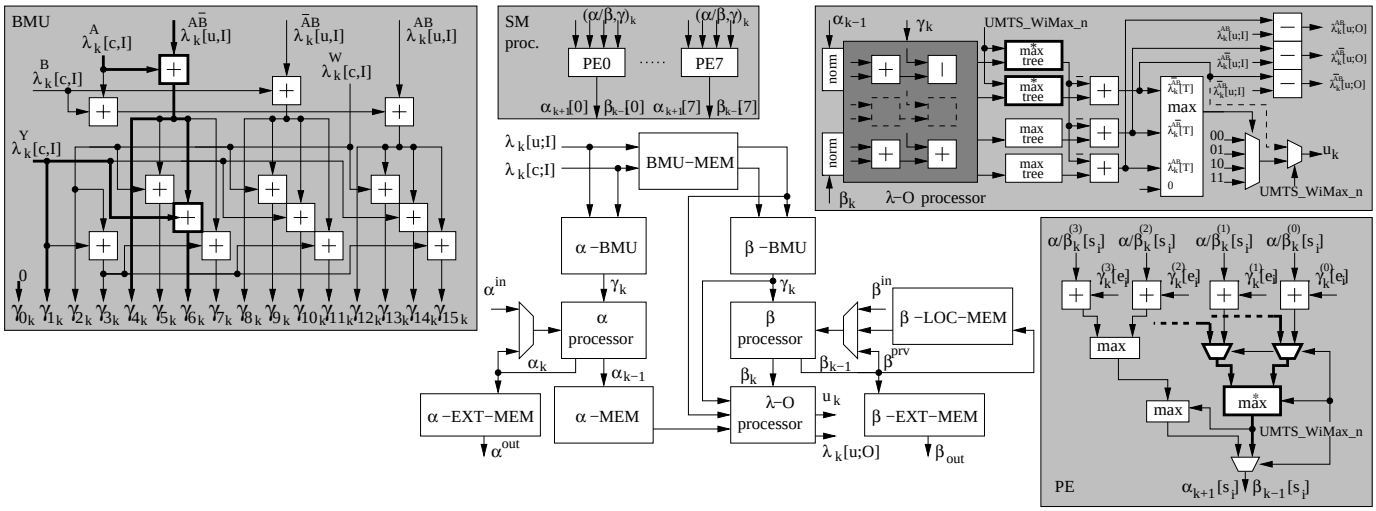


Figure 3. Flexible UMTS/WiMax SISO architecture: in the center the general SISO architecture and its building blocks, on the left the BMU structure, on the top the SM processor structure, on the right the $\lambda - O$ processor and the detail of the SM processor processing element (PE)

1p-LP in $\mu$m$^2$ as a function of the number of bits ($bit$) used to represent the LLRs or the LPs. The range explored in this analysis ($bit \in [4, 8]$) shows that the SISO-LLR complexity is slightly larger than that of the SISO-LP. However, the amount of memory required by a SISO-LP based decoder increases more than that of a SISO-LLR based one. In Fig. 2 the complexity of a single SISO decoder, including the interleaver, is also shown. Further experiments show that increasing $P$, the overhead required by the LP based decoder with respect to the LLR based one decreases from 7.6% ($P = 1$) to 2.2% ($P = 8$). However, the LLR based decoder is still less complex. As a consequence, in the following the LLR based decoder architecture is described according to the formalism introduced in section II. This choice implies that the BMU-MEM contains $W$ words, each word being made of 4 channel LLRs ($\lambda_k[c; I]$) represented on $n_{\lambda c}$ bits and 3 extrinsic LLRs ($\lambda_k[u; I]$) represented on $n_{\lambda u}$ bits.

## IV. Low complexity interleaver design

Since the proposed architecture achieves the throughput required by UMTS with a single SISO, the UMTS interleaver parallelization is not addressed in this work. In order to reduce as much as possible the complexity of the UMTS permutation generator, we implemented the two step architecture detailed in [18], which is very similar to that proposed in [19].

On the other hand a parallel decoder ($P = 4$) is required to achieve the WiMax throughput with the assumed clock frequency, $f_{clk}$=200 MHz; as a consequence we designed the parallel interleaver shown in Fig. 4. The permutation algorithm specified in the WiMax standard is structured in two steps. The first step switches $\lambda_k^{A\overline{B}}[u(e)]$ and $\lambda_k^{\overline{A}B}[u(e)]$ stored at odd addresses leaving $\lambda_k^{AB}[u(e)]$ un-moved (where $\lambda_k[u(e)]$ can be either $\lambda_k[u(e), I]$ or $\lambda_k[u(e), O]$). The second step provides the interleaved address $i$ of the $j$-th $\lambda_j[u(e)]$ triplet as

$$i = (P_0 \cdot j + P_j^{'}) \bmod N_c \qquad (9)$$

where $P_j^{'} \in \{1, (1 + N_c/2 + P_1), (1 + P_2), (1 + N_c/2 + P_3)\}$ and $P_j$ are constants depending on $N_c$, defined in [2] and [18]. The interleaver architecture can be simplified by rewriting (9) as $i = [(P_0 \cdot j) \bmod N_c + (P_j^{'} \bmod N_c)] \bmod N_c$ as detailed in [18].

In this work we consider that the throughput sustained by the decoder scales with $N_c$, namely for short block lengths a single SISO is active (e.g. $T = 6.8$Mb/s with $N_c = 24$). When $192 \leq N_c < 480$, two SISOs are active (e.g. $T = 30$Mb/s with $N_c = 192$) and when $960 \leq N_c \leq 2400$ all the four SISOs are active (e.g. $T = 78.95$Mb/s with $N_c = 960$). Given $P$ SISOs and $P$ memories to interleave extrinsic information, two different SISOs should not read from or write to the same memory at the same time to avoid collisions. As detailed in [18], the resulting parallel interleaver with variable parallelism degree is a circular shifting interleaver [20], whose implementation requires nearly the same complexity as the non-parallel version. In fact, the collision free characteristic is achieved by making the SISOs accessing at the same time the same location of different memories.

## V. Flexible UMTS/WiMax SISO architecture

In the following paragraphs the solutions employed to share the SISO architecture between UMTS and WiMax are detailed. In Fig. 3 the logic employed in the UMTS mode is highlighted with bold lines into the WiMax SISO building blocks.

*a) BMU sharing - left side Fig. 3:* due to the trellis symmetry, the WiMax BMU computes 16 possible BMs, whereas the UMTS BMU only 4. Thus, starting from the equations required to compute the 8 WiMax $\alpha_k$ and $\beta_k$ SMs, we can reduce the number of multiplexers to implement the UMTS SMs. Since in the UMTS mode (UMTS_WiMax_n = '1') 12 outputs of the BMU would not be used, we properly replicate the 4 possible UMTS BMs so that the 2×16 input adders in the $\alpha$ and $\beta$ processor do not require to be multiplexed.

*b) SMs processors sharing - bottom right side Fig. 3:* $\alpha_k^{(j)}[s_i]$ and $\beta_k^{(j)}[s_i]$ are the $j$-th SMs connected to the $i$-th state ($s_i$) at the $k$-th trellis step and $\gamma_k^{(j)}[e_i]$ the corresponding BMs. The UMTS mode at the $k$-th trellis step requires the

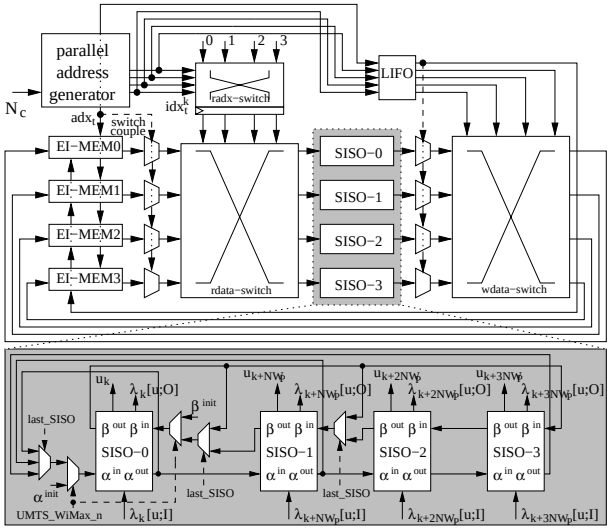


Figure 4. WiMax parallel interleaver architecture, in the shaded part SISO connection and inter SISO SM exchange network are highlighted

$i$-th PE to combine 2 SMs and 2 BMs to produce a new SM, whereas 4 SMs and 4 BMs are required in the WiMax mode. The $\overset{*}{\max}$ function shared by the UMTS and the WiMax decoder is implemented as a programmable two input $\overset{*}{\max}$. Since the UMTS turbo code achieves excellent performance even with a 1 bit correction (3 positions LUT [16]), the 1 bit correction can be exploited to substitute the last adder in the standard Add-Compare-Select-Offset implementation with a simpler, programmable increment.

*c) $\lambda - O$ processor sharing - top right side Fig. 3:* the $\lambda - O$ processor input stage is made of two normalization blocks (norm) devoted to subtracting the 0-state ($s_0$) from the others, $\alpha_k(s_i) - \alpha_k(s_0)$ and $\beta_k(s_i) - \beta_k(s_0)$.

The normalized SMs, combined with $\gamma_k$, become the inputs of the $\overset{*}{\max}$ trees (two $\overset{*}{\max}$ trees for UMTS and four $\overset{*}{\max}$ trees for WiMax). The $\overset{*}{\max}$ tree output referred to $\tilde{u}$ is subtracted from the others; then, subtracting the corresponding $\lambda_k[u;I]$, the output extrinsic LLRs are obtained. To ease the decoded bits generation the hard decision circuit is embedded into the $\lambda - O$ processor. For a binary turbo decoder it can be implemented taking the sign of $\lambda_k[T] = \lambda_k[u;I] + \lambda_k[u;O]$. On the other hand for a duo-binary turbo decoder the hard decision is selected as the couple with maximum LLR in $\{0, \lambda_k^{\overline{AB}}[T], \lambda_k^{A\overline{B}}[T], \lambda_k^{AB}[T]\}$, as shown in Fig. 3.

*d) SM exchange network sharing:* to grant a windowed elaboration, the $\alpha$-MEM contains $W$ words, each word being made of 8 SMs, represented on $n_{SM}$ bits. As stated in section III the SISO complexity and latency can be reduced [17] implementing a $\beta$ metrics inheritance strategy at the expenses of additional memory. Given the number of windows per SISO ($NW_P$), a $NW_P$-1 words local memory ($\beta$-LOC-MEM) stores the SMs at the boundary of two consecutive windows ($\beta^{prv}$). Each word is made of 8 SMs, each of which is represented on $n_{SM}$ bits. Moreover, every SISO requires two 8 SMs values to initialize its trellis portion ($\alpha^{in}$ and $\beta^{in}$). This architecture is suited to a single SISO decoder, where only intra SISO SMs inheritance is required. However, in a parallel SISO decoder, inter SISO SMs inheritance is required to properly initialize trellis slices of different SISOs. This can be achieved by inserting two 2-position shift registers ($\alpha$-EXT-MEM and $\beta$-EXT-MEM) to exchange the $\alpha^{out}$ and $\beta^{out}$ SMs with the neighboring SISOs. As depicted in Fig. 4, a simple network allows to properly exchange the boundary SMs among the different SISOs considering that in the UMTS mode the trellis starting and termination SMs are fixed ($\alpha^{init}$ and $\beta^{init}$) whereas in the WiMax mode they are estimated as explained in section III. Depending on which is the last SISO active (last_SISO) the SMs ought to be inherited from a different SISO.

Table I
TURBO DECODER ARCHITECTURES COMPARISON: SYMBOL MODE (S) CAN BE BINARY (B) OR DUO-BINARY (D), CMOS TECHNOLOGY PROCESS (TP), LOGIC (L), MEMORY (M), CLOCK FREQUENCY ($f_{clk}$) AND THROUGHPUT ($T$)

| Architecture | S | TP [$\mu$m] | L [kgate] | M [kbit] | $f_{clk}$ [MHz] | $T$ [Mb/s] |
|---|---|---|---|---|---|---|
| [24] | B | 0.18 | 410 | 450 | 145 | 24 |
| [25] | B | 0.18 | 121 | - | 285 | 27.6 |
| [22] | B | 0.5 | 75 | 390 | 95 | 3.8 |
| **This work** | **B** | **0.13** | **75** | **70.9** | 200 | 12 |
| [23] | D | - | ∼480 | ∼713 | ∼200 | - |
| **This work** | **D** | **0.13** | **171** | **133.6** | 200 | 90.36 |
| [7]-I | B/D | 0.09 | 97 | - | 335 | 7.4 |
| [7]-II | B/D | 0.09 | 1552 | - | 335 | 100 |
| **This work** | **B/D** | **0.13** | **204** | **148.6** | **200** | 90.36 |

## VI. IMPLEMENTATION, THROUGHPUT AND LATENCY

According to the literature [16], [21], $n_{\lambda c}$= 6, $n_{\lambda u}$= 8 and $n_{SM}$ = 12 have been chosen as a significant, conservative case for both UMTS and WiMax. Synthesis results on a 0.13 $\mu$m standard cell technology show that the proposed, flexible UMTS/WiMax architecture requires about 204 kgates. The single mode WiMax architecture requires about 171 kgates, the UMTS one described in [22], similar to the one employed in this work, requires 75 kgates. So the combination of the two single mode decoders leads to 246 kgates: the proposed solution is 17.1% less complex. As stated in section III, memory sharing and on-the-fly generation of scrambled addresses grant a large area saving. This is confirmed by the actual memory requirements: the $P = 4$ WiMax decoder requires 133.6 kbits, whereas the UMTS decoder requires 70.9 kbits. As a consequence the two architectures require 204.5 kbits. The proposed solution with memory sharing requires only 148.6 kbits, thus it grants a memory saving of 27.3% and a total area saving of 27.7% compared to the two single mode architectures. In Table I the proposed architecture is compared to some binary and duo-binary turbo decoder architectures. The proposed dual mode architecture shows excellent performance and complexity figures compared both to a fixed implementation [23] and to a programmable solution [7] ([7]-I refers to the single processor solution, whereas [7]-II is related to the 16 processor architecture).

As it can be inferred from Table I the proposed architecture achieves a throughput higher than specified by the WiMax standard. This implies that enough processing power is available for the concurrent execution of the UMTS and WiMax

Table II
THROUGHPUT/LATENCY

| Architecture | $T$ [Mb/s] | $D$ [$\mu$s] |
|---|---|---|
| Single standard UMTS | 12 | 414 |
| Single standard WiMax | 90.36 | 53 |
| Concurrent UMTS | 2 | 2400 |
| Concurrent WiMax | 75 | 2400 |

decoding. Of course external buffers must be available to receive an UMTS frame while WiMax decoding is in progress and viceversa. In the following we prove that the proposed architecture can support the concurrent execution of the UMTS and WiMax decoding. The time required to decode $K$ blocks of $N$ bits with the proposed architecture is

$$\Theta_x(K) = K\frac{N}{T_x} \tag{10}$$

where $T_x$ is the throughput of the proposed architecture and $x$ can be either $U$ for UMTS or $W$ for WiMax. As a consequence

$$\Theta_{tot}(K) = \Theta_W(K) + \Theta_U(1) \tag{11}$$

is the total time required to decode the hybrid sequence of $K$ WiMax blocks and 1 UMTS block. Concurrent decoding can be sustained only if the throughputs required ($\Phi$) by UMTS and WiMax are achieved for maximum block length:

$$K\frac{N_W}{\Theta_{tot}} \geq \Phi_W \quad \frac{N_U}{\Theta_{tot}} \geq \Phi_U \tag{12}$$

where $N_W$=4800 (WiMax) and $N_U$=5114 (UMTS). Substituting (10) and (11) in (12) we obtain

$$\left\lceil \frac{\Phi_W T_W}{T_U(T_W - \Phi_W)} \right\rceil \leq K \leq \left\lfloor \frac{T_W}{\Phi_U} - \frac{T_W}{T_U} \right\rfloor = K_M \tag{13}$$

The final choice for $K$ has been made taking $\Phi_U$=2 Mb/s and solving (13) for the maximum $\Phi_W$; this results in $K_M$=37 and $\Phi_W$=75 Mb/s. The concurrent decoding also affects latency. The proposed architecture latency can be obtained from (8) as

$$D = \frac{2I(\tilde{N} + SISO_l P)}{P f_{clk}} \tag{14}$$

namely $D_U \simeq$414 $\mu$s and $D_W \simeq$53 $\mu$s. Thus, the total latency to decode $K_M$ WiMax blocks and 1 UMTS block is

$$D_{tot} = \frac{2I}{f_{clk}}\left[\frac{K_M N_c}{P} + N + SISO_l(1 + K_M)\right] \tag{15}$$

In the worst case ($N_c$=2400 and $N$=5114) we obtain $D_{tot} \simeq 2.4$ ms, which is a small percentage of the global latency specified by both UMTS and WiMax standards. The single and multi-standard figures of throughput and latency offered by the presented architecture are summarized in Table II.

## VII. CONCLUSION

In this paper a flexible UMTS/WiMax turbo decoder architecture has been presented together with a parallel WiMax interleaver architecture. Compared to a single mode, parallel WiMax architecture the proposed one exhibits a limited complexity overhead. Moreover, compared to a separated dual mode UMTS/WiMax turbo decoder architecture, it achieves the 17.1% logic reduction and the 27.3% memory reduction.

## REFERENCES


[1] "Universal mobile telecommunications system (UMTS), multiplexing and channel coding (TDD)," Mar. 2007, 3GPP TS 25.222 version 7.2.0.

[2] "IEEE Std 802.16, part 16: air interface for fixed broadband wireless access systems," Oct. 2004.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error correcting coding and decoding: Turbo codes," in *IEEE ICC*, 1993, pp. 1064–1070.

[4] C. Berrou, M. Jezequel, C. Douillard, and S. Kerouedan, "The advantages of non-binary turbo codes," in *IEEE Inf. Theory Workshop*, 2001, pp. 61–63.

[5] C. Zhan, T. Arslan, A. T. Erdogan, and S. MacDougall, "An efficient decoder scheme for double binary circular turbo codes," in *IEEE ICASSP*, 2006, pp. 229–232.

[6] S. Papaharalabos, P. Sweeney, and B. Evans, "Constant log-MAP decoding algorithm for duo-binary turbo codes," *IET Electronics Letters*, vol. 42, no. 12, pp. 709–710, Jun 2006.

[7] O. Muller, A. Baghdadi, and M. Jezequel, "ASIP-baser multiprocessor SOC design for simple and double binary turbo decoding," in *DATE*, 2006, pp. 1330–1335.

[8] T. Vogt and N. Wehn, "A reconfigurable applcation specific instruction set processor for Viterbi and Log-MAP decoding," in *IEEE Workshop on Signal Proc. Systems Design and Implementation*, 2006, pp. 142–147.

[9] M. C. Shin and I. C. Park, "SIMD processor-based turbo decoder supporting multiple third-generation wireless standards," *IEEE Trans. on VLSI*, vol. 15, no. 7, pp. 801–810, Jul 2007.

[10] M. J. Thul, N. Wehn, and L. P. Rao, "Enabling high-speed turbo-decoding through concurrent interleaving," in *IEEE ISCAS*, 2002, pp. 897–900.

[11] F. Speziali and J. Zory, "Scalable and area efficient concurrent interleaver for high throughput turbo-decoders," in *IEEE Euromicro Symposium on Digital System Design*, 2004, pp. 334–341.

[12] A. Tarable, L. Dinoi, and S. Benedetto, "Design of prunable interleavers for parallel turbo decoder architectures," *IEEE Comm. Letters*, vol. 11, no. 2, pp. 167–169, Feb 2007.

[13] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Soft-input soft-output modules for the construction and distributed iterative decoding of code networks," *European Trans. on Telecom.*, vol. 9, no. 2, pp. 155–172, Mar/Apr 1998.

[14] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. on Inf. Theory*, vol. 20, no. 3, pp. 284–287, Mar 1974.

[15] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the Log domain," in *IEEE ICC*, 1995, pp. 1009–1013.

[16] G. Montorsi and S. Benedetto, "Design of fixed-point iterative decoders for concatenated codes with interleavers," *IEEE Journal on Selected Areas in Comm.*, vol. 19, no. 5, pp. 871–882, May 2001.

[17] A. Abbasfar and K. Yao, "An efficient and practical architecture for high speed turbo decoders," in *IEEE VTC*, 2003, pp. 337–341.

[18] M. Martina, M. Nicola, and G. Masera, "Low complexity UMTS and WiMax interleavers design," Politecnico di Torino, Dipartimento di Elettronica, Tech. Rep., 2007, available at: http://www.vlsilab.polito.it/∼martina.

[19] Z. Wang and Q. Li, "Very low-complexity hardware interleaver for turbo decoding," *IEEE Trans. on Circuits and Systems II*, vol. 54, no. 7, pp. 636–640, Jul 2007.

[20] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and nonrandom permutations," *TDA Progress Report*, vol. 42-122, pp. 56–65, Aug 1995.

[21] A. P. Hekstra, "An alternative to metric rescaling in Viterbi decoders," *IEEE Trans. on Comm.*, vol. 37, no. 11, pp. 1220–1222, Nov 1989.

[22] G. Masera, G. Piccinini, M. Ruo-Roch, and M. Zamboni, "VLSI architectures for turbo codes," *IEEE Trans. on VLSI*, vol. 7, no. 3, pp. 369–379, Sep 1999.

[23] A. Bartolazzi, G. Cardarilli, A. Del-Re, D. Giancristofaro, and M. Re, "Implementation of DVB-RCS turbo decoder for satellite on-board processing," in *IEEE Intern. Conf. on Circuits and Systems for Comm.*, 2002, pp. 142–145.

[24] M. Bickerstaff, L. Davis, C. Thomas, D. Garrett, and C. Nicol, "A 24Mb/s radix-4 LogMAP turbo decoder for 3GPP-HSDPA mobile wireless," in *IEEE Intern. Solid State Circuits Conf.*, 2003, pp. 150–151.

[25] S. J. Lee, N. R. Shanbhag, and A. C. Singer, "A 285-MHz pipelined MAP decoder in 0.18-$\mu$m CMOS," *IEEE Jour. of Solid-State Circuits*, vol. 40, no. 8, pp. 1718–1725, Aug 2005.